In the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

1.   (Canceled)

1   2.   (Previously Presented) A software development method for
2   debugging software on a target system having a plurality of
3   processors configured with shared memory in which a software memory
4   bus performs processing of shared memory access requests using a
5   method for transparently writing to shared memory when debugging a
6   multiple processor system, the method comprising the steps of:
7   receiving user input to define a hardware configuration of the
8   target system;
9   creating a software memory map of how each of the plurality of
10   processors may access memory including whether each of said
11   plurality of processors may read from and write to a range of
12   memory or may only read from said range of memory;
13   loading drivers for each of the plurality of processors;
14   activating a first debug session associated with a first
15   processor of the plurality of processors;
16   activating at least a second debug session associated with a
17   second processor of the plurality of processors wherein each debug
18   session is operable to transmit read requests and write requests to
19   its associated processor;
20   processing shared memory access requests via a software memory
21   bus;
22   detecting a write request to a shared memory location by the
23   first debug session; and
24   if the first processor associated with the first debug session
25   has write access to the shared memory location
26   then

−2−

27   selecting the first processor to perform the write

28   request;

29   else performing the following steps a-c:

30   a.   searching the software memory map to determine

31   if the second processor has write access to the shared memory

32   location;

33   b.   selecting the second processor to perform the

34   write request; and

35   c.   passing the write request initiated by the first

36   debug session to the selected processor for execution.

1   3.   (Original) The method of Claim 2 wherein the step of

2   passing the write request comprises the steps of:

3   searching the software memory map for a second plurality

4   of processors;

5   broadcasting the write request to the second plurality of

6   processors; and

7   performing cache coherency updates in response to the

8   write request in each of the second plurality of processors.

1   4.   (Original) The method of Claim 3 wherein the step of

2   broadcasting the write request comprises indicating that the write

3   request is intended for maintaining cache coherency as opposed to a

4   normal write request.

1   5.   (Original) The method of Claim 3 wherein the step of

2   performing comprises using cache coherency capabilities, if any, of

3   a processor in response to the write request intended for

4   maintaining cache coherency.

1   6.   (Previously Presented) The method of Claim 3 wherein:

- 3 -

2   the step of creating comprises denoting in the software memory

3 map all the shared memory locations that contain program

4 instructions upon each initialization of the target system;

5   the step of passing the write request additionally comprises

6 the step of determining that the shared memory location contains a

7 program instruction; and

8   the cache is an instruction cache.

   7 to 12.   (Canceled)

1   13.   (Previously Presented) A software development method for

2 debugging software on a target system having a plurality of

3 processors configured with shared memory, comprising steps of:

4   receiving user input to define a hardware configuration of the

5 target system;

6   creating a software memory map of how each of the plurality of

7 processors may access memory including whether each of said

8 plurality of processors may read from and write to a range of

9 memory or may only read from said range of memory;

10   loading drivers for each of the plurality of processors;

11   activating a first debug session associated with a first

12 processor of the plurality of processors;

13   activating at least a second debug session associated with a

14 second processor of the plurality of processors wherein each debug

15 session is operable to transmit read requests and write requests to

16 its associated processor; and

17   processing shared memory access requests via a software memory

18 bus, in which processing of shared memory access requests via the

19 software memory bus uses a method for transparently maintaining

20 cache coherency when debugging a multiple processor system with

21 common shared memory, the method comprising the steps of:

— 4 —

22   denoting in the software memory map the shared memory
23 locations whether or not each processor of the plurality of
24 processors has a cache;
25   detecting a write request to a shared memory location by
26 the first debug session;
27   passing the write request initiated by the first debug
28 session to the first processor for execution;
29   searching the software representation of the memory map
30 for a first plurality of processors that have read access to the
31 shared memory location;
32   broadcasting the write request to the first plurality of
33 processors; and
34   performing cache coherency updates in response to the
35 write request in each of the first plurality of processors.

1   14.   (Original) The method of Claim 13 wherein the step of
2 broadcasting the write request comprises indicating that the write
3 request is intended for maintaining cache coherency as opposed to a
4 normal write request.

1   15.   (Original) The method of Claim 13 wherein the step of
2 performing comprises using cache coherency capabilities, if any, of
3 a processor in response to the write request intended for
4 maintaining cache coherency.

1   16.   (Currently Amended) a A method for transparently
2 maintaining cache coherency when debugging a multiple processor
3 system with common shared instruction memory, the method comprising
4 the steps of:
5   if the a first processor of the multiple processor system
6 associated with the a first debug session has write access to the a
7 shared memory location

— 5 —

8   then

9   selecting the first processor to perform ~~the~~ a write
10   request;

11   else performing the following steps a-b:

12   a.   searching ~~the~~ a software memory map for a second
13   processor <u>of</u> the <u>multiple</u> <u>processor</u> <u>system</u> with write access to the
14   shared memory location;

15   b.   selecting the second processor to perform the
16   write request;

17   passing the write request initiated by the first debug session
18   to the selected processor for execution;

19   searching the software memory map for a second plurality of
20   processors <u>of</u> the <u>multiple</u> <u>processor</u> <u>system</u> that have read access
21   to the shared memory location;

22   broadcasting the write request to the second plurality of
23   processors; and

24   performing cache coherency updates in response to the write
25   request in each of the second plurality of processors.

1   17.   (Original) The method of Claim 16 wherein the step of
2   broadcasting the write request comprises indicating that the write
3   request is intended for maintaining cache coherency as opposed to a
4   normal write request.

          18 and 19.   (Canceled)

– 6 –